Multi-Label Answer Aggregation for Crowdsourcing

Nguyen Thanh Tam[#], Huynh Huu Viet[*], Nguyen Quoc Viet Hung[*],
Matthias Weidlich[†], Hongzhi Yin[*], Xiaofang Zhou[*]

[#] *École Polytechnique Fédérale de Lausanne* [*] *The University of Queensland* [†] *Humboldt-Universität zu Berlin*

ABSTRACT

Crowdsourcing has been widely established as a means to enable human computation at large scale, in particular for tasks that require manual labelling of large sets of data items. Answers obtained from heterogeneous crowd workers are aggregated to obtain a robust result. However, existing methods for answer aggregation assume that answers are given as a single label per item. Hence, these methods are ineffective for common multi-labelling problems such as image tagging and document annotation, where items are assigned sets of labels. In this paper, we propose a novel Bayesian nonparametric model for multi-label answer aggregation. It enables us to predict labels for non-grounded items, while taking into account dependencies between the labels in different answer sets. We also show how this model is instantiated for incremental learning, incorporating new answers from crowd workers as they arrive. An evaluation of our method using a number of large-scale, real-world crowdsourcing datasets reveals that it consistently outperforms the state-of-the-art in answer aggregation in terms of precision, recall, and robustness against faulty workers and data sparsity.

1. INTRODUCTION

Fuelled by the massive availability of Internet users, crowdsourcing has been widely established as a means for human computation at large-scale [24]. Tasks that are rather trivial for humans, but computationally expensive or even unsolvable for machines can be efficiently addressed by crowdsourcing. Specifically, crowdsourcing has been applied for such diverse applications as data acquisition [2], data integration [37], data mining [30], information extraction [12], and information retrieval [35]. Today, a large number of platforms, such as Amazon Mechanical Turk and CrowdFlower, facilitate the development of crowdsourcing applications.

Aggregation of Crowd Answers. Most crowdsourcing setups are based on questions (aka tasks) that, once posted to crowdsourcing platform, are answered by users (aka crowd workers) for financial rewards. Yet, each task is answered by multiple workers to accommodate for their different levels of expertise and motivation [17]. Aggregation of answers obtained for a single task shall complement individual errors, exploiting the 'wisdom of the crowd'.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

Answer aggregation is challenging for several reasons. The worker population may contain faulty workers (e.g., spammers) that give random answers, but are hard to identify before-hand in the absence of detailed worker information. Furthermore, workers may be unintentionally biased by personal interest or systematic misunderstanding of the tasks [34]. Aggregation of answers is also complicated by limited mutual information between workers and tasks, e.g., some workers are assigned with too few tasks and vice-versa [36]. To overcome these challenges, various methods for automatic answer aggregation have been proposed in the literature (see [15] for survey), including (i) non-iterative techniques which compute the aggregated answer as a linear combination of votes, and (ii) iterative techniques which leverage mutual reinforcing relations between workers and answers.

Multi-label Answer Aggregation. The above aggregation methods have been developed for single-label tasks comprising of a set of labels and a set of items—a crowd worker is expected to assign a single label to each item. This assumption, however, does not hold for many crowdsourcing applications that received much attention recently, such as text categorization, image classification, and medical diagnosis [1, 6, 22]. These applications define multi-label tasks, where workers shall provide a set of labels per item.

Despite the increased noise and bias due to the freedom to choose multiple labels per item, multi-label answer aggregation is inherently more complex than its single-label counterpart. First, the labels obtained as part of different answers are often correlated. For instance, in movie classification, movies about a *super-hero* are often associated with the genre *action* [28]. Construction of a correct set of labels needs to deal with the exponential growth of combinations of labels and dependencies between them. Second, workers no longer either agree or disagree on the answer to a question, but consensus becomes partial. As a consequence, it becomes difficult to assess the reliability of workers since they may provide supposedly correct and incorrect labels at the same time.

Approach. In this paper, we propose a Bayesian nonparametric model in order to capture the distinct properties of multi-label answer aggregation. That is, co-occurrence dependencies between labels are represented by the notion of latent label clusters. This notion is motivated by the observation that items can be often be grouped together, if they share similar labels. Furthermore, partial consensus between workers is modelled by grouping together workers with similar answers. This enables us to construct an aggregated answer based on the consensus of groups of workers instead of consensus among individuals. The resulting model, called *Clustering-based Bayesian Combination of Multi-label Classifiers* (cBCMC), generalises models developed for single-label answer aggregation [20] and enables incremental learning using stochastic variational inference [13].

Contribution and Structure. Our contributions along with the structure of the paper can be summarized as follows:

Problem Setting (§2) We motivate the need for multi-label answers aggregation. We further elaborate on types of crowd workers, formalize the problem stetting, and outline requirements for solutions to the problem of multi-label answer aggregation. Further, we discuss the limitations of the state-of-the-art in using Bayesian nonparametric models for answer aggregation.

Novel Model for Multi-Label Answer Aggregation (§3) We present a generative model for multi-label answer aggregation, called *Clustering-based Bayesian Combination of Multi-label Classifiers (cBCMC)*. Specifically, we show how to perform model inference (finding the probability distribution of model parameters given information on worker answers or true labels) and model instantiation (estimating item labels based on the given information and the inferred parameter distributions).

Methods for Incremental Computation (§4) To cope with the continuous arrival of new answers in a crowdsourcing setting, we present methods for incremental computation. We show how model inference is facilitated by updating the model parameters based on new data instead of inferring a model from scratch.

Evaluation (§5) Experiments with real-world datasets highlight the effectiveness of the proposed cBCMC model, consistently outperforming the state-of-the-art in terms of precision, recall, and robustness against faulty workers. When using incremental learning, we observe speed-ups of up to 17× in runtime, with only moderate reduction in label prediction accuracy.

Finally, §6 reviews related work, before §7 concludes the paper.

2. PROBLEM SETTING

We first introduce a motivating example for multi-label answer aggregation and elaborate on challenges induced by different types of crowd workers. Then, we present a problem statement and discuss requirements for potential solutions. Finally, we review the state-of-the-art in answer aggregation against these requirements.

2.1 Motivating Example

We consider an image tagging task, in which workers assign one or more labels to a picture. For simplicity, these labels are encoded by numbers from 1 to 5. Table 1 illustrates an exemplary crowdsourcing result, in which five workers (u_1 - u_5) provided their answers to four pictures (i_1 - i_4). The correct, yet generally unknown, label assignment is shown in a separate column.

Table 1: Answers provided by five workers for four pictures

	u_1	u_2	u_3	u_4	u_5	Correct	Majority [4]
i_1	{4,5}	{4,5}	{4}	{1}	{5}	{5}	{4,5}
i_2	{2,3}	{1,4}	{4}	{2}	{3,4}	{3,4}	{4}
i_3	{1,2}	{4}	{4}	{3}	{4,5}	{4,5}	{4}
i_4	{1,2}	{2,3}	{4}	{4}	{1,2,3}	{1,2,3}	{2}

1: sky, 2: plane, 3: sun, 4: water, 5: tree

Answer aggregation computes, for each item, a joint answer based on the input provided by workers. A common method to derive an aggregated answer is majority voting [4], which considers all labels separately. If the ratio of 'votes' from workers for a label is larger than 0.5, the respective label is included in the aggregation result. Compared to the actually correct assignment, the result obtained in this case has two issues, though: (i) it is partially incorrect (e.g., label 4 is not correct for i_1), and (ii) partially incomplete (e.g. labels 1 and 3 shall also be assigned to i_4).

These issues have two main causes. First, workers are considered equally, whereas it is well known that they have different characteristics. Previous studies [17] identified different types of crowd workers: (1) Reliable workers have deep knowledge about specific domains and answer questions with high reliability; (2) Normal workers have general knowledge to give correct answers, but make mistakes occasionally; (3) Sloppy workers have little knowledge and often give wrong answers unintentionally; (4) Uniform spammers intentionally give the same answer for all questions; (5) Random spammers give random answers for all questions. For binary classification tasks, these types can be described by the two-coin model [39]. As illustrated in Fig. 1, it assesses worker quality in terms of sensitivity (the proportion of positives that are correctly classified) and specificity (the proportion of negatives that are correctly classified).

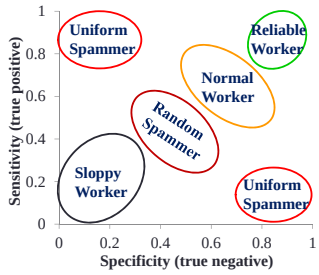


Figure 1: Characterization of worker types

In the above example, u_3 is a uniform spammer, assigning the same labels to all pictures. Yet, these answers are reflected in aggregated result. Removing u_3, for instance, yields the correct result for picture i_1. Worker u_4 is a random spammer, whereas the remaining workers can be classified as truthful (u_5) or normal (u_1 and u_3). In practice, different worker types are frequently encountered. A recent study [39] reported on a population consisting of 38% spammers, 18% sloppy, 16% normal, and 27% reliable workers for a binary classification task.

A second cause for the issues observed in the example is the neglect of dependencies between labels. For instance, label 2 often co-occurs with labels 3 and 1. Such correlation can be useful in the aggregation. If we also include label 1 and label 3 whenever label 2 has been assigned, for instance, the obtained result would be correct for picture i_4 when using majority voting.

2.2 Problem Statement

We capture the setting of multi-label answer aggregation by a set of workers $\mathcal{U}$, identified by their indices, $\mathcal{U} \triangleq \{1,\ldots,U\}$ that provide answers for a set of items $\mathcal{N}$, also identified by their indices, $\mathcal{N} \triangleq \{1,\ldots,I\}$. $\mathcal{Z} \triangleq \{1,\ldots,C\}$ is the set of all possible labels for these items. Each answer by a crowd worker is a subset of $\mathcal{Z}$. Formally, answers are modelled as an $I \times U$ *answer matrix*:

$$\mathcal{M} \triangleq \begin{pmatrix} x_{11} & \ldots & x_{1U} \\ \ldots & \ldots & \ldots \\ x_{I1} & \ldots & x_{IU} \end{pmatrix}$$

where $x_{iu} \subseteq \mathcal{Z}$ is the set of labels assigned to item i by worker u, or $x_{iu} = \emptyset$ if worker u has not provided an answer for item i.

PROBLEM 1. *Given a set of items $\mathcal{N}$, a set of workers $\mathcal{U}$, a set of labels $\mathcal{Z}$, and an answer matrix $\mathcal{M}$, the problem of multi-label answer aggregation is the construction of a* deterministic assignment $d : I \rightarrow 2^{\mathcal{Z}}$ *assigning a set of labels to each item.*

A baseline solution to the above problem is to construct the assignment d by majority voting, as illustrated above. Yet, observing the issues that stem from the application of majority voting, we derive a set of requirements that shall be met by any approach to answer aggregation in order to be useful in the multi-label setting.

(R1) *Consideration of worker communities:* In practice, there is little control over the selection of crowd workers. Answer aggregation, thus, shall capture and characterize worker behaviours, to assess the likelihood of them providing correct answers and to justify their effects in the aggregated result.

(R2) *Support for partial answer validity:* Against the background of diverse worker types and their distribution in practice, the correctness of answers shall be assessed in a fine-granular manner, i.e., at the level of individual labels. This is a prerequisite to make efficient use of normal and sloppy workers in particular.

(R3) *Exploitation of label dependencies:* In many multi-label settings, similar items are assigned overlapping sets of labels. Such dependencies between labels, e.g., their co-occurrence in the answers provided by crowd workers, shall be exploited to improve the soundness and completeness of answer aggregation.

(R4) *Adaptivity of aggregation model:* The characteristics of a crowdsourcing application (e.g., the number of worker communities) may vary over time upon the arrival of new data. This requires dynamic adaptation of the aggregation model to reflect the evolving relations between the obtained answers.

2.3 State-of-the-Art in Answer Aggregation

Taking the aforementioned requirements as a starting point, it turns out that most existing algorithms for aggregating crowd answers are inapplicable, see [15] for a comprehensive evaluation. The vast majority of aggregation methods do not incorporate diverse characteristics of workers and their implications for answer correctness, and thus fail to address requirement (R1).

A notable exception is *Clustering Based Bayesian Combination of Classifiers* (cBCC) as recently proposed by Moreno et al. [20], which is a Bayesian nonparametric generative model [9]. In general, grounding answer aggregation in a probabilistic model has the advantage that complex interactions can be established between workers, answers, items, and labels, all modelled as random variables. Probabilistic techniques enable computation of the certainty of label assignments to predict the labels of non-grounded items.

The cBCC model takes into account worker communities (R1), by a notion of worker clusters that group together workers based on their trustworthiness and domain knowledge. In contrast to methods that evaluate individual workers, e.g., by means of confusion matrices [25], models that rely on clusters of workers are less prone to errors when data is sparse. This makes them particularly suited for crowdsourcing where each item is processed only by a fraction of the worker population due to budget constraints.

Since cBCC is a generative model, it supports self-configuration through inference and prediction, i.e. the more information is available, the more accurate the inferred model parameters and the estimated labels of remaining items are. Since the model is also nonparametric, the number of parameters is adjusted to the data, thereby enabling adaptivity of the aggregation model (R4). The Bayesian property of the model helps to reduce over-fitting by inferring probability distribution over random variables instead of singleton values. In addition, Bayesian models are well suited to cope with online settings—new information can be encoded into posterior distributions used in the inference and prediction process.

The above observations highlight the advantages of Bayesian nonparametric models for answer aggregation in crowdsourcing. However, the sole model of this class presented in the literature so far, the cBCC model, is applicable only in the single-label setting. Neither does it support partial answer validity (R2) nor can it exploit label dependencies (R3). Therefore, we develop a new Bayesian nonparametric model that generalises the ideas behind cBCC and is tailored to the multi-label setting.

3. MULTI-LABEL ANSWER AGGREGATION IN CROWDSOURCING

This section introduces our novel model for multi-label answer aggregation, referred to as *Clustering Based Bayesian Combination of Multi-label Classifiers* (cBCMC). We first give an intuitive overview of the model, before we turn to its formalisation. Then, we outline the application of cBCMC for multi-label answer aggregation: we derive a scalable inference method with Variational Bayes and show how to predict the labels of non-grounded items.

3.1 Overview of the Approach

To address the problem of multi-label answer aggregation, we consider each element of the given answer matrix as an observed random variable. The true labels of each item are also modelled as a random variable. While a few of them may be observed (e.g., due to test questions [19]), the vast majority of these variables are unobserved. To predict the value of these unobserved variables, i.e., to estimate the labels for an item for which the true labels are not available, our cBCMC model adopts the generative process followed also in cBCC. All random variables are generated from parametrised probability distributions and the respective parameters are inferred from the observed variables. Here, worker communities are considered by a clustering of workers that is modelled nonparametrically by a Chinese Restaurant Process (CRP). In second step, the values of the unobserved variables are predicted.

Being based on a Bayesian nonparametric model and following a generative process, our cBCMC model satisfies the outlined requirements regarding the consideration of worker communities (R1) and adaptivity (R4) precisely as discussed for cBCC in §2.3. The specific challenges of answer aggregation in the multi-label setting, in turn, are addressed as follows.

Dependencies between labels (R3) are incorporated in our model by clustering items in the answer aggregation process. Items in a cluster are assumed to be similar and, thus, be assigned the same set of labels. The latter implicitly encodes dependencies between labels in terms of co-occurrence.

To support partial validity of answers (R2), we follow the intuition that obtaining a label for an item can be seen as randomly selecting labels of the respective item cluster, given a worker community. Hence, we model the labels as being generated from a Multinomial distribution over the item clusters and worker communities. Since this is a random process, workers in the same community may still provide different labels for items of the same cluster.

3.2 The Model of cBCMC

The input of multi-label answer aggregation (Problem 1) is a set of items $\mathcal{N} \triangleq \{1,\ldots,I\}$, a set of workers $\mathcal{U} \triangleq \{1,\ldots,U\}$, a set of labels $\mathcal{Z} \triangleq \{1,\ldots,C\}$, all identified by the indices of their elements, and an answer matrix $\mathcal{M}$. We define the model of Clustering Based Bayesian Combination of Multi-label Classifiers (cBCMC) as follows (notations are summarised in Table 2). All non-empty answers in $\mathcal{M}$ are modelled as observed variables $\boldsymbol{x} \in (2^{\mathcal{Z}})^{I\times U}$, where x_{iu} denotes the set of labels assigned to item i by worker u. Further, $y \in (2^{\mathcal{Z}})^{I}$ are random variables modelling the true labels of all items. True labels may be known for some items, which is captured by a set of observed variables $\boldsymbol{y} \subseteq y$. In general, $\boldsymbol{y}$ may be empty. The values of variables $\boldsymbol{x}$ and $\boldsymbol{y}$ can be represented as a C-dimensional vector, such that each of its components is set to one, if the respective label is present. Thus, we can consider the observed values of $\boldsymbol{x}$ and $\boldsymbol{y}$ as samples from a Multinomial distribution.

Worker communities, item clusters, and label selection are modelled as follows (Fig. 2 shows a graphical representation):

Worker Communities. There is a finite set of worker communities π, identified by indices, $\pi \triangleq \{1,\dots,M\}$, that partition the set of workers and are not known in advance. The (unknown) assignment of workers to communities is captured by a set of random variables $z \in \pi^U$, such that z_u denotes the community of worker u. We generate π nonparametrically using a Chinese Restaurant Process (CRP). Following [20], it can be interpreted as the induced distribution over the partition space by a Dirichlet Process [9]. Technically, if π follows a CRP distribution, $\pi \sim \mathrm{CRP}(\alpha)$, the samples from this prior follow the following distributions

$$p(z_u = m \mid z_{-u}, \pi, \alpha) \propto \begin{cases} n_m^{-u} & \text{if } \exists\, z_{u'} \in z_{-u} : z_{u'} = m \\ \alpha & \text{otherwise} \end{cases}$$

where $z_{-u} = z \setminus \{z_u\}$ and n_m^{-u} is the number of elements in z_{-u} with community m. As shown by Sethuraman [29], π can be constructed using a stick-breaking process as follows. Let π'_m, $m = 1,2,3,\dots$ be sampled from a Beta distribution $\mathrm{Beta}(1,\alpha)$. Then, the community proportion π_m is calculated using the above sticks π'_m, such that

$$\pi_1 = \pi'_1,\ \dots\ , \pi_m = \pi'_m \prod_{j=1}^{m-1} \left(1 - \pi'_j\right), \cdots \tag{1}$$

When conducting inference, we will only estimate the stick distribution π' since the original distribution π can be calculated directly from π' as above. We further note that the nonparametric approach generalises the extreme cases. If M tends to infinity, each worker is a single community (e.g., no two workers provide similar answers). If M tends to zero, all workers form a single community (e.g., only expert workers) and the result is similar to majority voting.

Item Clusters. To model clusters of similar items, which tend to get assigned the same sets of labels, we follow the approach introduced for worker communities. There is a finite set τ of clusters, identified by indices, $\tau \triangleq \{1,\dots,T\}$, that partition the set of items and are not known in advance. The (unknown) assignment of items to these clustered is captured by random variables $l \in \tau^I$, such that l_i denotes the cluster of item i. Again, τ is generated nonparametrically by a Chinese Restaurant Process, i.e., $\tau \sim \mathrm{CRP}(\varepsilon)$.

The assignment of sets of labels to item clusters is modelled as a generation from a Multinomial distribution. For cluster t, this distribution is parameterised by $\phi_t \triangleq \{\phi_{t,1},\dots,\phi_{t,C}\}$, where $\phi_{t,c}$ is the probability that a given item in cluster t will have the label c. Items in a cluster may have different true labels as a result of the generating random process—yet, being in the same cluster, they are similar and thus share the labelling probabilities.

Label Selection. We model the labels obtained from workers as being generated from a Multinomial distribution over the labels of an item cluster, given a specific worker community. Each worker is characterised by a $C \times T$ confusion matrix ψ_m, where m is the community that the worker belongs to. We denote by ψ_m^t a column vector of C-dimensions, which contains the probabilities that a worker in community m assigns the respective labels given an item of cluster t. This model has the advantage that, instead of considering exponentially many subsets of labels, it is grounded in the number of all possible item clusters, which is tractable in practice.

For illustration, we consider a setting with four labels {1: girl, 2: boy, 3: dog, 4: cat}, two item clusters {1: people, 2: animal}, and two worker communities {1: trustworthy, 2: problematic}. Then, having a confusion matrix column vector $\psi_1^1 = [0.5, 0.5, 0, 0]$ means that workers of community 1 (trustworthy) assign an item of cluster 1 (people) the label 1 (girl), 2 (boy), 3 (dog), or 4 (cat) with probability 0.5, 0.5, 0, or 0, respectively.

Table 2: Overview of notations

$\mathcal{N}$	Set of I items, identified by indices, $\mathcal{N} \triangleq \{1,\dots,I\}$
$\mathcal{U}$	Set of U workers, identified by indices, $\mathcal{U} \triangleq \{1,\dots,U\}$
$\mathcal{Z}$	Set of C possible labels, identified by indices, $\mathcal{Z} \triangleq \{1,\dots,C\}$
$\mathcal{M}$	$I \times U$ Answer matrix
π	Set of M worker communities, identified by indices, $\pi \triangleq \{1,\dots,M\}$
τ	Set of T item clusters, identified by indices, $\tau \triangleq \{1,\dots,T\}$
z_u	Community of worker u
x_{iu}	Labels assigned to item i by worker u
ψ_m^t	Label assignment probabilities of workers in community m for items in cluster t
l_i	Cluster of item i
y_i	True labels assigned to item i
ϕ_t	Label assignment probabilities for items in cluster t

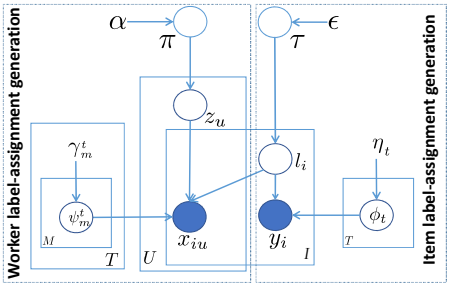


Figure 2: Graphical representation of the cBCMC model

Generative Process. Let Cat and Multi be Categorical and Multinomial distributions, respectively. Then, the generative process for the cBCMC model is defined as follows:

(1) For each item in $\mathcal{N}$ (right-hand side of Fig. 2):
   a) Generate the cluster for each item: $l_i \mid \tau \sim \mathrm{Cat}(\tau)$
   b) Generate the labels for each item from the cluster: $y_i \mid l_i, \phi \sim \mathrm{Mult}(\phi_{l_i})$

(2) For each worker in $\mathcal{U}$ (left-hand side of Fig. 2):
   a) Generate the community for each worker: $z_u \mid \pi \sim \mathrm{Cat}(\pi)$
   b) Generate the set of assigned labels for each worker and item from the labels of the item cluster and the confusion matrix of the worker's community: $x_{iu} \mid z_u, l_i, \psi \sim \mathrm{Mult}\left(\psi_{z_u}^{l_i}\right)$

Model Parameters. The cBCMC model is nonparametric since the number of worker communities in π and the number of item clusters in τ are not known in advance—they change with more observations ($\boldsymbol{x}$ and $\boldsymbol{y}$) becoming available.

In a Bayesian setting, we use the following priors for the parameters related to the worker communities and item clustering (Dir being a Dirichlet distribution):

$$\pi \sim \mathrm{CRP}(\alpha) \qquad \psi_m^t \sim \mathrm{Dir}\left(\gamma_m^t\right)$$
$$\tau \sim \mathrm{CRP}(\varepsilon) \qquad \phi_t \sim \mathrm{Dir}(\eta_t)$$

with $1 \le t \le T$ and $1 \le m \le M$. Both τ and π are unknown.

3.3 Inference

Inferring the parameters of the cBCMC model is, in fact, the estimation of values of the above priors $(\alpha, \varepsilon, \gamma, \eta)$. This is equivalent to inferring the posterior distribution of the unobserved variables $(\pi, \tau, z, l, \psi, \phi)$ under the observed variables $(\boldsymbol{x}, \boldsymbol{y})$, which is $p(\pi, \tau, z, l, \psi, \phi \mid \boldsymbol{x}, \boldsymbol{y})$, or $p(\pi', \tau', z, l, \psi, \phi \mid \boldsymbol{x}, \boldsymbol{y})$ using the stick-breaking representation for π and τ (see Eq. 1).

Approaches for (approximate) inference for statistical models can be classified into simulation methods and deterministic variational methods. The use of simulation such as Markov Chain Monte Carlo is problematic when applied to large-scale data sets

since convergence cannot be predicted. Thus, we resort to variational inference. Specifically, we propose a novel scalable method that follows the principles of variational Bayesian inference (VB).

In variational inference, instead of computing the posterior distribution directly, we infer an approximation $q(\pi', \tau', z, l, \psi, \phi)$, referred to as variational distributions:

$$q(\pi', \tau', z, l, \psi, \phi) = q(\pi' \mid \rho)\, q(\tau' \mid \upsilon) \prod_{u=1}^{U} q(z_u \mid \kappa_u) \prod_{i=1}^{I} q(l_i \mid \varphi_i) \prod_{m=1}^{M} \prod_{t=1}^{T} q(\psi_m^t \mid \lambda_m^t) \prod_{t=1}^{T} q(\phi_t \mid \zeta_t)$$

where $q(z_u \mid \kappa_u)$ and $q(l_i \mid \varphi_u)$ are M-dimensional and T-dimensional Multinomial distributions; and $q(\psi_m^t \mid \lambda_m^t)$ and $q(\phi_t \mid \zeta_t)$ are C-dimensional Dirichlet distributions.

For the variational distributions $q(\pi' \mid \rho)$ and $q(\tau' \mid \upsilon)$ we rely on a stick-breaking truncation representation for a Chinese Restaurant Process similar to those in [3], which are truncated to M and T, respectively. The variational distributions are:

$$q(\pi' \mid \rho) = \prod_{m=1}^{M-1} \text{Beta}(\pi'_m \mid \rho_{m1}, \rho_{m2})$$
$$q(\tau' \mid \upsilon) = \prod_{t=1}^{T-1} \text{Beta}(\tau'_t \mid \upsilon_{t1}, \upsilon_{t2})$$

To approximate the posterior distributions p by variational distributions q, we use the KL-divergence between them, $KL(q \mid p)$. With $\Theta \triangleq \{\pi', \tau', z, l, \psi, \phi\}$, it is defined as:

$$\begin{aligned} KL(q \mid p) &\triangleq -\int q(\Theta) \ln \frac{p(\Theta \mid \boldsymbol{x}, \boldsymbol{y})}{q(\Theta)} d\Theta \\ &= -\int q(\Theta) \ln \frac{p(\Theta, \boldsymbol{x}, \boldsymbol{y})}{q(\Theta)} d\Theta + \ln p(\boldsymbol{x}, \boldsymbol{y}) \\ &\geq -\int q(\Theta) \ln \frac{p(\Theta, \boldsymbol{x}, \boldsymbol{y})}{q(\Theta)} d\Theta \triangleq -\mathcal{L}(\Theta) \end{aligned}$$

$\mathcal{L}(\Theta)$ is called *evidence lower bound* (ELBO) and denotes the variational objective function. Using variational theory [16], taking derivatives of this lower bound with respect to each variational parameter, we derive the following coordinate ascent updates [3].

Local Updates. We first update local variables (connected to a single data point), i.e., z and l in our model. We update the respective distributions $q(z_u \mid \kappa_u)$ and $q(l_i \mid \varphi_i)$ as follows (details on the computation of these equations are given in Appendix A):

$$\kappa_{um} \propto \exp\left(\sum_{i=1}^{I} \sum_{t=1}^{T} \varphi_{it} \mathbb{E}\left[\ln p\left(x_{iu} \mid \psi_m^t\right)\right] + \mathbb{E}\left[\ln \pi_m\right] \right) \tag{2}$$

$$\varphi_{it} \propto \exp\left(\mathbb{E}\left[\ln p\left(y_i \mid \phi_t\right)\right] + \mathbb{E}\left[\ln \tau_t\right]\right) \tag{3}$$

Global Updates. Next, we consider the updates for global (or outer) variables (connected to multiple data points), i.e., π, τ, ψ, and ϕ in our model. We update $q(\pi' \mid \rho)$ and $q(\tau' \mid \upsilon)$ by means of:

$$\rho_{m1} = 1 + \sum_{u=1}^{U} \kappa_{um} \qquad \rho_{m2} = \alpha + \sum_{u=1}^{U} \sum_{l=m+1}^{M} \kappa_{ul} \tag{4}$$

$$\upsilon_{t1} = 1 + \sum_{i=1}^{I} \varphi_{it} \qquad \upsilon_{t2} = \varepsilon + \sum_{i=1}^{I} \sum_{l=t+1}^{T} \varphi_{il} \tag{5}$$

Here, $\alpha, \varepsilon > 0$ are 'prior beliefs' on the actual number of worker communities and item clusters. Yet, their effects are marginal, as the updates are dominated by the observed information (κ and φ).

Distributions $q(\psi_m^t \mid \lambda_m^t)$ and $q(\phi_t \mid \zeta_t)$ are update by means of:

$$\lambda_{mc}^t = \lambda_{m0}^t + \sum_{i=1}^{I} \varphi_{it} \sum_{u=1}^{U} \kappa_{um} x_{iu} \tag{6}$$

$$\zeta_{tc} = \zeta_{t0} + \sum_{i=1}^{I} \varphi_{it} y_i \tag{7}$$

We summarize our inference algorithm to learn the model parameters in Algorithm 1. It iteratively updates local parameters (κ, φ) and global parameters $(\rho, \upsilon, \lambda, \zeta)$. The observed data $(\boldsymbol{x}, \boldsymbol{y})$ is used in the updates of these parameters whenever their associated variables are connected to the observed variables. Note that many updates of variables are independent, which can be exploited to scale up the performance. For instance, the individual updates of κ parameters and φ parameters can be parallelised.

Algorithm 1 Variational Inference for the cBCMC Model

Input : Worker answers $\boldsymbol{x}$ and known true labels $\boldsymbol{y}$
Output: Estimated model parameters $\lambda, \zeta, \rho, \upsilon, \kappa, \varphi$

```
Random initialisation of λ, ζ, ρ, υ, κ, φ
while not converged do
    // Update the local variables
    for u ← 1,...,U and m ← 1,...,M do Update κ_um using Eq. 2 ;
    for i ← 1,...,I and t ← 1,...,T do Update φ_it using Eq. 3 ;
    // Update the global variables
    for m ← 1,...,M do
        Update ρ_m1 and ρ_m2 using Eq. 4
        for t ← 1,...,T and c ← 1,...,C do Update λ^t_mc using Eq. 6 ;
    for t ← 1,...,T do
        Update υ_t1 and υ_t2 using Eq. 5
        for for c ← 1,...,C do Update ζ_tc using Eq. 7 ;
return λ, ζ, ρ, υ, κ, φ
```

3.4 Prediction

To solve the multi-label answer aggregation problem, we construct a deterministic assignment $d : I \to 2^C$ using the maximum likelihood principle (MAP) [7]. After approximating the values of model parameters $\mathcal{P} \triangleq \{\alpha, \varepsilon, \gamma, \eta\}$, we predict the labels of non-grounded items. Technically, given an item i, we denote by $\boldsymbol{x}_{U_i} \triangleq \{x_{vi} \mid v \in U_i\}$ the labels assigned by the workers U_i who provided answers for this item. Further, $\mathcal{D} \triangleq \{\boldsymbol{x}, \boldsymbol{y}\}$ denotes the assigned labels as well as known labels as used in the inference. We now compute y_i using MAP estimation of the probability $p(y_i \mid \boldsymbol{x}_{U_i}, \mathcal{D}, \mathcal{P})$:

$$y_i^* = \underset{y_i}{\text{argmax}}\, p(y_i \mid \boldsymbol{x}_{U_i}, \mathcal{D}, \mathcal{P}) = \underset{y_i}{\text{argmax}}\, p(y_i, \boldsymbol{x}_{U_i} \mid \mathcal{D}, \mathcal{P}) \tag{8}$$

since $p(y_i \mid \boldsymbol{x}_{U_i}, \mathcal{D}, \mathcal{P}) = p(y_i, \boldsymbol{x}_{U_i} \mid \mathcal{D}, \mathcal{P}) / p(\boldsymbol{x}_{U_i} \mid \mathcal{D}, \mathcal{P})$, there is no direct dependency between y_i and $\boldsymbol{x}_{U_i}$ in the graphical representation, and the divisor does not depend on y_i. The above formulation of the conditional probability of y_i, i.e. $p(y_i \mid \boldsymbol{x}_{U_i}, \mathcal{D}, \mathcal{P})$, has the advantage that it covers diverse crowdsourcing settings. For instance, the absence of known true labels ($\boldsymbol{y} = \emptyset$ in $\mathcal{D} = \{\boldsymbol{x}, \boldsymbol{y}\}$) or a separation of training data and testing data ($\boldsymbol{x}_{U_i} \nsubseteq \boldsymbol{x}$) can be directly encoded in this formulation.

To compute $p(y_i, \boldsymbol{x}_{U_i} \mid \mathcal{D}, \mathcal{P})$, we factorise over all probabilistic dependencies in the graphical model representation. Using the derivation outlined in Appendix C, we arrive at the following form:

$$\begin{aligned} &p(y_i, \boldsymbol{x}_{U_i} \mid \mathcal{D}, \mathcal{P}) \\ &= \sum_{t=1}^{T} \varphi_{it} \prod_{u \in U_i} \left(\sum_{m=1}^{M} \kappa_{um} p\left(x_{ui} \mid \psi_m^{(t)MAP}\right) \right) p\left(y_i \mid \phi_t^{MAP}\right) \end{aligned}$$

where $\psi_m^{(t)MAP}$ and ϕ_t^{MAP} are maximum a posteriori (MAP) estimates (aka modes) of the inferred distributions of ψ_m^t and ϕ_t.

However, the maximization problem in Eq. 8 is a zero-one integer problem, which is known to be NP-hard—the exhaustive search needs to explore $2^C - 1$ combinations of labels. Against this background, we may use a greedy search algorithm to approximate the mode y_i^* of the above distribution. Initially, all elements y_{ic}^* of the vector y_i^* are set as zeros. Then, we proceed iteratively and, in each iteration, set to one the element y_{ic^*} that leads to the largest increase of $p\left(y_i^*, \boldsymbol{x}_{U_i} \mid \mathcal{D}, \mathcal{P}\right)$. This procedure terminates once $p\left(y_i^*, \boldsymbol{x}_{U_i} \mid \mathcal{D}, \mathcal{P}\right)$ can not be further increased. The final configuration of y_i^* will be the instantiation value for the deterministic assignment. Note that this instantiation can be done independently for all items, so that this step can be parallelised.

4. INCREMENTAL COMPUTATION

The inference and prediction methods introduced above for the cBCMC model solve the multi-label answer aggregation problem in a static setting. However, in many cases, tasks are not answered immediately when posted on a crowdsourcing platform. Rather, the set of worker answers is gradually building up over time and intermediate aggregation results are valuable from an application point of view [35]. For instance, intermediate results may indicate that a task is too difficult for workers, so that it shall be re-designed. Also, if intermediate results are of high quality, the crowdsourcing process can be terminated early to save cost.

We cater for such an online setting by means of incremental computation for the cBCMC model. We present an inference algorithm that incrementally updates the model parameters based on new data, which are then used for predicting the true labels of all items. In each learning iteration, we maintain only the most recent parameter values, thereby avoiding the cost of repeatedly building the model from the complete set of answers. While this approach comes with a modest reduction in aggregation quality (explored in our experiments), it greatly improves aggregation efficiency.

4.1 Incremental Learning with Stochastic Variational Inference

The deterministic variational inference presented in the previous section for the static setting maximises the EBLO function $\mathcal{L}(\boldsymbol{\Theta})$ using coordinate-ascent for each of the parameters of variational distributions. To realise incremental learning, we rely on stochastic variational inference [13] and apply stochastic optimization to the EBLO function based on newly received data.

Technically, data is received as a series of batches $b = 1, 2, \ldots$. Each batch b contains the answers of a fixed number of workers $\mathcal{U}_b$ (with U_b being the cardinality of $\mathcal{U}_b$) for a set of items $\mathcal{N}_b$. We consider new answers as a subsample and, based thereon, derive a stochastic gradient. Specifically, we compute the difference ∇ between old and new values of each parameter. Following [13, 33], we classify variational distributions as being *global* or *local*. In our setting, $q(l_i \mid \varphi_i)$, $q(\pi' \mid \rho)$, $q(\tau' \mid \upsilon)$, $q(\psi_m^t \mid \lambda_m^t)$, and $q(\phi_t \mid \zeta_t)$ are global, whereas $q(z_u \mid \kappa_u)$ is local (φ now becomes global as we consider multiple items in one update).

Natural Gradients. For the local distribution, we reuse the update formulation given for VB inference (i.e., Eq. 2). The respective distribution is connected to a single data point, which can be computed directly from the new data. In contrast, for the global distributions, natural gradients are obtained for each variable over all $u \in \mathcal{U}_b$ as follows (the derivation can be found in Appendix B):

$$\nabla_{\lambda_m^t} \mathcal{L}_u = \frac{-\lambda_m^t + \gamma_m^t + U \sum_{i \in \mathcal{N}_b} \varphi_{it} \kappa_{um} x_{iu}}{U} \tag{9}$$

$$\nabla_{\zeta_t} \mathcal{L}_u = \frac{-\zeta_t + \eta + \sum_{i \in \mathcal{N}_b} \varphi_{it} y_i}{U} \tag{10}$$

$$\nabla_{\rho_{m1}} \mathcal{L}_u = \frac{-\rho_{m1} + 1 + U \kappa_{um}}{U} \tag{11}$$

$$\nabla_{\rho_{m2}} \mathcal{L}_u = \frac{-\rho_{m2} + \alpha + U \sum_{l=m+1}^{M} \kappa_{um}}{U} \tag{12}$$

$$\nabla_{\upsilon_{t1}} \mathcal{L}_u = \frac{-\upsilon_{t1} + 1 + \sum_{i \in \mathcal{N}_b} \varphi_{it}}{U} \tag{13}$$

$$\nabla_{\upsilon_{t2}} \mathcal{L}_u = \frac{-\upsilon_{t2} + \varepsilon + \sum_{l=t+1}^{T} \sum_{i \in \mathcal{N}_b} \varphi_{il}}{U} \tag{14}$$

The natural gradient for $q(l_i \mid \varphi_i)$ is difficult to compute since the mean-parameterisation requires the constraints $\sum_{t=1}^{T} \varphi_{it} = 1$ and $0 \leq \varphi_{it} \leq 1$ to be satisfied. Hence, we prefer to work with a minimal canonical parameterisation in exponential family form, parametrising the distribution by μ instead of φ:

$$q(l_i \mid \mu_i) = \exp\left(\langle \mu_i, S(l_i) \rangle - B(\mu_i)\right)$$

where $\mu_i = \left[\mu_{i1}, \ldots \mu_{i(T-1)}\right]^\mathsf{T}$ is a $T-1$-dimensional vector parameter, $B(\boldsymbol{\mu}_i) = 1 + \sum_{t=1}^{T-1} \exp(\mu_{it})$ is a normalisation function, and $S(l) = [\mathbf{I}(l-1), \ldots, \mathbf{I}(l-T+1)]^\mathsf{T}$ is a sufficient statistic function. The idea of sufficient statistics is to only maintain the minimal/sufficient information instead of all data points to compute the probability distribution. In our case, the sufficient function is defined as a $T-1$-dimensional binary vector, containing a value of one only at position l. That is, $\mathbf{I}(x)$ is an indicator function, $\mathbf{I}(x) = 1$ if $x = 0$; and $\mathbf{I}(x) = 0$, otherwise. The natural gradient for parameter μ is:

$$\nabla_{\mu_{it}} \mathcal{L}_u = \frac{-\mu_{it} + \mathbb{E}[\varepsilon_t] - \mathbb{E}[\varepsilon_T] + U(a_{it} - a_{tT})}{U} \tag{15}$$

where $a_{it} = \sum_{m=1}^{M} \kappa_{um} \mathbb{E}[\ln p(x_{iu} \mid \psi_m^t)]$ for $t = 1, \ldots, T$. To derive φ from μ, we use the following transformation:

$$\varphi_{it} = \frac{\exp(\mu_{it})}{1 + \sum_{t=1}^{T-1} \exp(\mu_{it})} \quad \text{for } t = 1, \ldots T-1 \tag{16}$$

$$\varphi_{iT} = \frac{1}{1 + \sum_{t=1}^{T-1} \exp(\mu_{it})} \tag{17}$$

Learning Rate. In incremental learning, a learning rate ω_b needs to be specified as a function of the batch index b. To ensure the convergence of the gradients, ω_b shall satisfy two conditions:

$$\sum_{b=1}^{\infty} \omega_b = \infty \quad \text{and} \quad \sum_{b=1}^{\infty} \omega_b^2 < \infty.$$

In general, a too small value of ω_b might lead to non-optimality; a too large value might lead to non-convergence. The learning rate actually depends on r, aka the forgetting rate. If r is large, ω_b becomes small, and the old parameter values are only slightly changed. Finding an appropriate value for r is specific to a dataset. As suggested in [13], we vary $r \in (0.5, 1]$ in our experiments.

Online Updates. Using the above gradients, the updates of all necessary parameters in the online setting become:

$$\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \omega_b \frac{U}{U_b} \sum_{u \in \mathcal{U}_b} \nabla_{\boldsymbol{\lambda}} \mathcal{L}_u \qquad \boldsymbol{\zeta} \leftarrow \boldsymbol{\zeta} + \omega_b \frac{U}{U_b} \sum_{u \in \mathcal{U}_b} \nabla_{\boldsymbol{\zeta}} \mathcal{L}_u \tag{18}$$

$$\boldsymbol{\rho} \leftarrow \boldsymbol{\rho} + \omega_b \frac{U}{U_b} \sum_{u \in \mathcal{U}_b} \nabla_{\boldsymbol{\rho}} \mathcal{L}_u \qquad \boldsymbol{\upsilon} \leftarrow \boldsymbol{\upsilon} + \omega_b \frac{U}{U_b} \sum_{u \in \mathcal{U}_b} \nabla_{\boldsymbol{\upsilon}} \mathcal{L}_u \tag{19}$$

$$\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} + \omega_b \frac{U}{U_b} \sum_{u \in \mathcal{U}_b} \nabla_{\boldsymbol{\mu}} \mathcal{L}_u \tag{20}$$

The algorithm for incremental learning for the cBCMC model is illustrated in Algorithm 2. In each iteration, a pre-defined number of answers are collected from the crowd. Based on the new data, we compute the natural gradients and update the model parameters.

Algorithm 2 Stochastic Variational Inference for the cBCMC model

Input : Continuously updated worker answers $\boldsymbol{x}$ and known true labels $\boldsymbol{y}$
Output: Estimated model parameters $\lambda, \zeta, \rho, \upsilon, \kappa, \varphi$

Random initialisation of $\lambda, \zeta, \rho, \upsilon, \kappa, \varphi$
$b \leftarrow 1$ // The batch index
while *more answers are available* **do**
  Fetch the b-th batch of answers of users $\mathcal{U}_b$ for items $\mathcal{N}_b$ and set $b \leftarrow b+1$
  // Update the local variables
  for $u \in \mathcal{U}_b$ *and* $m \leftarrow 1, \ldots, M$ **do** Update κ_{um} using Eq. 2. ;
  // Update the global variables
  Compute the natural gradients using Eq. 9 to 15.
  Set learning rate $\omega_b = (1+b)^{-r}$
  Update $\lambda, \zeta, \rho, \upsilon, \mu$ using Eq. 18 to 20.
  Compute φ using Eq. 16 to 17.
return $\lambda, \zeta, \rho, \upsilon, \kappa, \varphi$

4.2 Online Prediction

Online prediction enables us to perform the instantiation of labels incrementally, upon the arrival of new answers. Different from the inference procedure for incremental learning, online prediction does not compute the difference between the old and new labels assignments. The reason is that the most recent parameter values constitute the probability distributions of all data obtained so far. Each time new answers are obtained, the parameter values are updated and their values can be used to generate the corresponding approximated posterior distributions of model variables required for instantiation, i.e., $q^{(b)}(l_i \mid \varphi_i)$, $q^{(b)}(z_u \mid \kappa_u)$, $q^{(b)}(\psi_m^t \mid \lambda_m^t)$, $q^{(b)}(\phi_t \mid \zeta_t)$, $q^{(b)}(\pi' \mid \rho)$ and $q^{(b)}(\tau' \mid \upsilon)$, where $b = 1, 2, \ldots$ is the batch index. These posteriors are approximations of their offline counterparts and, thus, are used as input of the instantiation procedure in §3.4.

5. EVALUATION

We evaluated our approach to multi-label answer aggregation along several dimensions. We first elaborate on our experimental setup (§5.1), before evaluating the following aspects:

- The effectiveness of the cBCMC model for answer aggregation in a static setting (§5.2).
- The effectiveness of the cBCMC model when using incremental computation in an online setting (§5.3).
- The efficiency of incremental computation (§5.4).
- The importance of representing worker communities and item clusters in the cBCMC model (§5.5).

5.1 Experimental Setup

Task Design. Aiming at a realistic evaluation setup, we follow best practices on task design for crowdsourcing:

Batch processing: Each task consists of multiple items that are to be labelled by a single user. To mediate the trade-off between the overhead of switching tasks and the cognitive load of a single task, we follow recent studies on crowdsourcing effectiveness [14], suggesting a task size of 10 items.

Pricing: We vary the price for a task over the datasets based on the difficulty of the respective tasks. Considering that a simple task would take five minutes to complete and that the average wage of workers is around 2.00$/h [27], we set the task price to 0.1$, 0.2$, and 0.3$ for simple, medium, and difficult tasks, respectively.

Datasets. Our experiments have been conducted using five real-world datasets, spanning diverse application scenarios:

(1) Image annotation: From the NUS-WIDE set of tagged web images [6], we randomly selected 2000 images, such that tags are uniformly distributed. Each image has up to 10 tags, which serve as ground truth in our experiments. Workers were asked to assign a subset of 81 possible tags to each image.

Table 3: Statistics for real-world datasets

Quantity	Dataset				
	(1) image	(2) topic	(3) aspect	(4) entity	(5) movie
# Items	269,648	16M	3710	2400	500
# Labels	81	49	262	1450	22
# Questions	2000	2000	3710	2400	500
# Workers	416	313	482	517	936
# Answers	22920	15080	19780	15510	14430
Unit Price ($)	0.01	0.02	0.03	0.02	0.01

(2) Topic annotation: We relied on a random sample of 2000 Twitter messages from the collection that was used in the TREC 2011 Microblog track [22]. This dataset assigns up to five topics (from a set of 49 topics) to each tweet and, again, we ensured a uniform distribution of topic labels in our sample.

(3) Aspect extraction: This dataset is about assigning evaluation aspects (e.g., *price* or *menu*) to restaurant reviews [11]. The ground truth, provided by [23], assigns up to five aspects (out of 262) to 3710 reviews. We designed crowdsourcing tasks that asked workers to assign a subset of 20 possible labels to each review. The set of possible labels contains the true labels and is filled up with the labels that have the highest co-occurrences with the true labels.

(4) Entity extraction: The T-NER dataset [26] contains 2400 tweets, to which entities (of ten categories such as products or facilities) shall be assigned. It also includes the ground truth for all tweets. We asked workers to tag each word of a tweet as being an entity or non-entity, so that each tweet is assigned a set of entities.

(5) Movie tagging: This dataset has been created by crawling the IMDB website, randomly selecting 500 movies from a total of 22 genres. As such, the ground truth directly stems from the IMDB website. Workers have been asked to assign genres to these movies.

We employed workers to perform item labelling using the CrowdFlower platform [31]. In total, we spent a total budget of 8772 tasks for all datasets and ended up having a repository of 87720 label annotations for 10610 items from 2664 users, see also Table 3.

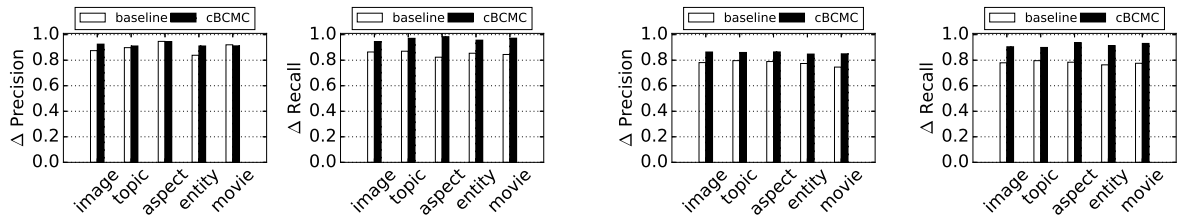
The resulting datasets cover diverse crowdsourcing scenarios: the distribution of worker answers is skewed in datasets (1) and (5), whereas it is normal in (3); tasks in datasets (2), (3), and (4) require understanding of unstructured text, which is more difficult than the tasks in (1) and (5); labels in (1), (2), and (4) are strongly correlated, whereas there is little correlation between labels in (5).

Metrics. In multi-label answer aggregation, results can be partially correct. We therefore rely on the set-based definition of precision and recall to evaluate the individual correctness of each item. Per item i, *individual precision* P_i is the ratio of correctly predicted labels and the total number of predicted labels, whereas *individual recall* R_i is the ratio of correctly predicted labels and the total number of true labels. For a complete dataset, *precision* P and *recall* R are the respective averages over all items. With $Y_i \triangleq \bigcup_{j \in \mathcal{Z}} \{j \mid y_{ij} = 1\}$ and $Y_i^* \triangleq \bigcup_{j \in \mathcal{Z}} \{j \mid y_{ij}^* = 1\}$, the measures are defined as:

$$P_i \triangleq \frac{|Y_i \cap Y_i^*|}{|Y_i^*|} \quad R_i \triangleq \frac{|Y_i \cap Y_i^*|}{|Y_i|} \quad P \triangleq \sum_{i=1}^{I} \frac{P_i}{I} \quad R \triangleq \sum_{i=1}^{I} = \frac{R_i}{I}$$

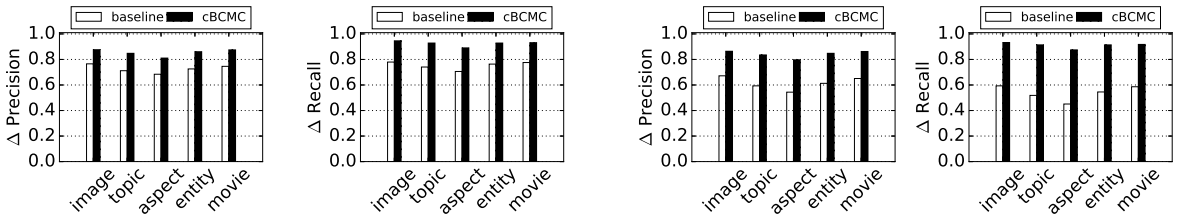
Baseline. We compare our approach against a baseline using majority voting, which is the only available aggregation method for the multi-label setting [4]. To ensure a fair comparison, we adapt the standard approach (see §2) by injecting knowledge on the result cardinality. That is, the baseline ranks labels per item by their number of appearances in the answers. We then consider the top-k labels, with k being the number of labels assigned by our approach (ties are resolved randomly and we average of multiple solutions).

Experimental Environment. All experimental results have been obtained on an Intel Core i7 system (3.4GHz, 12GB RAM), using parallelisation (as mentioned in §3 and §4) whenever possible.



(a) Sparsity level = 25% **(b)** Sparsity level = 50%

Figure 3: Effects of sparsity (compared to performance at sparsity level = 0% as ratio)



(a) Spammer ratio = 10% **(b)** Spammer ratio = 20%

Figure 4: Effects of spammers (compared to performance at spammer ratio = 0% as ratio)

5.2 Effectiveness in a Static Setting

Accuracy. We first evaluate the accuracy of our approach based on the cBCMC model against the baseline method in a static setting. That is, we measure individual precision and individual recall of each item of the aforementioned five datasets.

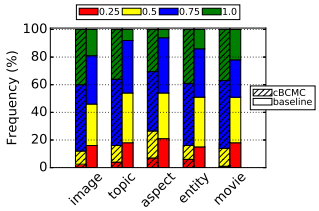


Figure 5: Individual Precision

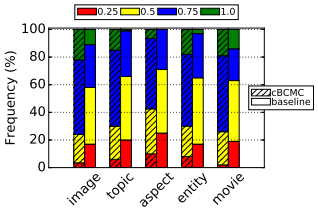


Figure 6: Individual Recall

Fig. 5 and 6 show the obtained results in terms of histograms over all items, where the bins are $[0, 0.25]$, $(0, 25, 0.5]$, $(0.5, 0.75]$, and $(0.75, 1]$. We observe that our approach outperforms the baseline methods for all datasets: there are notably more items with high individual precision and recall (bins $(0.5, 0.75]$ and $(0.75, 1]$) in all cases. We further observe that the absolute number of items for which we obtain accurate results varies across the datasets: *aspect* tasks are difficult, which leads to the lowest precision and recall values. For dataset *image*, we observe better results than for *movie*, since we can exploit co-occurrence dependencies between labels.

Robustness against Sparsity. In crowdsourcing scenarios, the answer matrix is typically sparse: most workers process only a few of the items of a particular application. We investigate the effect of sparsity on aggregation accuracy by randomly removing a certain share of the answers, leaving 25% or 50% of the data per dataset. We then measure precision and recall, averaged over 100 runs.

As illustrated in Fig. 3, in general, precision and recall decrease if answers are removed. However, answer aggregation based on our cBCMC model is affected less by data sparsity compared to the baseline method. For instance, for *image* tasks, when removing half of the input data (sparsity level 50%), the precision of our method is already 86% of the precision obtained using all answers. The baseline, in turn, achieves only 78% of the precision obtained using all answers in this case. This effect is due to the notion of worker communities in the cBCMC model that help to identify consistent answers for an item even if it was processed only by a few workers.

Robustness to Spammers. As discussed in §2, crowdsourcing applications suffer from faulty workers, such as random and uniform spammers, which can account for up to 40% of the worker population [8, 32]. Even though we may be able to detect different types of workers (based on their characteristics), the predicted labels may be incorrect, since faulty answers can be dominating. We investigate this aspect by adding answers of spammers to the datasets, such that they account for 10% or 20% of the data.

As expected, the results in Fig. 4 show that precision and recall decrease when spammers are included. However, our approach is much less affected by spammers as is the baseline method, in particular for large amounts of spammers (20%). For example, for the *aspect* dataset, the precision of the baseline method decreases from 0.49 to 0.43, whereas it stays nearly constant with our approach, achieving 0.71 and 0.70, respectively. This highlights that our approach can not only detect communities of spammers, but also limits their influence on the aggregation result.

5.3 Effectiveness in an Online Setting

Incremental computation for the cBCMC model as introduced in §4 aims at increasing the efficiency of computation. Yet, it may come at the expense of decreased effectiveness, i.e., lower accuracy in terms of precision and recall. We therefore compare the accuracy of the baseline method with the cBCMC model, once with the inference mechanisms for a static setting (*offline*) and once with the approach with incremental learning (*online*). To this end, we simulate an online setting by randomly selecting new worker answers to represent newly arriving data, in steps of 10% of the dataset size. For the non-incremental methods (baseline and offline), this setup corresponds to a step-wise increase of the sparsity level from 10% to 100% and the prediction is always based on the complete set of answers received so far.

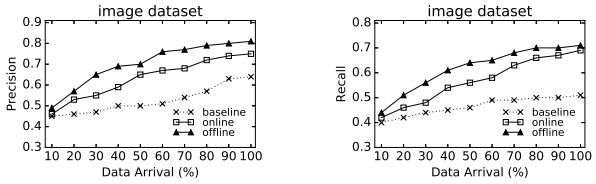


Figure 7: Effects of data arrival

The result for the *image* dataset is shown in Fig. 7. We notice that indeed, precision and recall are worse when using incremental computation for the cBCMC model. Yet, even with incremental computation, the results are significantly better than those of the baseline. For example, when the share of available answers increases from 10% to 50%, the precision of the cBCMC model with incremental computation increases from 0.49 to 0.70, while for the baseline, this increase is only from 0.45 to 0.50. This underlines that the summarised information about item clusters and worker communities maintained by our incremental inference method still enables more accurate aggregation compared to the baseline.

Table 4: Effects of data arrival (at 100%)

Dataset	Precision			Recall		
	baseline	online	offline	baseline	online	offline
image	0.64	0.75	0.81	0.59	0.69	0.74
topic	0.59	0.71	0.79	0.54	0.65	0.70
aspect	0.57	0.67	0.74	0.51	0.59	0.64
entity	0.62	0.70	0.79	0.55	0.64	0.70
movie	0.63	0.74	0.80	0.58	0.68	0.73

The result for the *image* dataset in Fig. 7 is representative for all datasets. Table 4 shows precision and recall obtained with the three methods after all answers have been processed. While incremental computation based on the cBCMC model incurs a moderate drop in accuracy compared to the non-incremental approach, it achieves consistently higher accuracy than the baseline method.

5.4 Efficiency of Incremental Computation

We now turn to the efficiency of incremental computation for the cBCMC model and measure the runtime of the inference mechanism for a static setting (*offline*) and the runtime of the incremental inference (*online*), in relation to the size of the input data. To have a controlled experimental setup, we generated a synthetic dataset, comprising 10,000 items and 500 workers. The actual input data has been generated randomly by the generative process of the cBCMC model, for which the priors are set based on the inference results for the five real-world datasets. We consider the average over these five configurations and over 100 experiment runs. In the experiment, we vary the density, taking between 5% and 30% of the complete answer matrix as input. Non-incremental inference is said to converge, if all model parameter differences in two consecutive inference iterations are below 10^{-3}. For incremental inference, we set the batch size to 50 answers and the forget rate to 0.8.

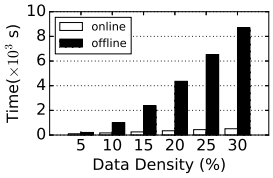


Figure 8: Runtime of inference mechanisms for the cBCMC model

As shown in Fig. 8, the incremental inference mechanism is indeed much more efficient than the non-incremental one. Inference in a static setting is non-linear: the computation requires iteration over two dimensions of the input data and an increased model complexity means that more iterations are needed to converge. Incremental inference, in turn, is performed on a fixed number of newly received answers and, therefore, scales linearly.

5.5 Importance of Model Aspects

Finally, we assess the importance of explicitly capturing worker communities and item clusters by comparing the accuracy of our cBCMC model with two simplified versions: *No_Z* removes the community structure (variable z) from the model, i.e., each worker is a singleton community; *No_L* removes the item cluster structure (variable l), i.e., each item represents a singleton cluster. However, the *No_L* model turned out to be intractable for all except the *movie* dataset, since all possible subsets of labels have to be considered.

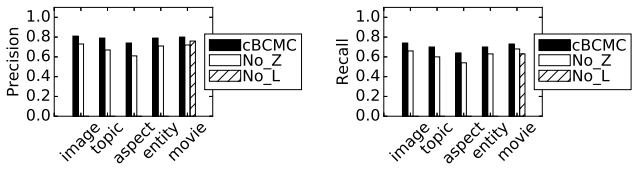


Figure 9: Effects of model aspects

Fig. 9 shows that the cBCMC model consistently achieves the highest precision and recall. Improvements over the *No_Z* model are particularly large for the more difficult datasets (*topic* and *aspect*), since differentiation of workers is effective in these cases.

We further note that the *No_Z* model achieves higher precision, but lower recall than the *No_L* model. This highlights that worker communities help to improve correctness by identifying faulty workers, whereas item clusters improve completeness by exploiting label co-occurrence dependencies.

6. RELATED WORK

Having discussed the context of answer aggregation in crowdsourcing in §1 and §2, below, we focus on further related areas.

Multi-label Problems. Multi-label problems have been solved in related research fields, such as multi-label classification [38], ordinal classification [18], and data streams classification [10]. Multi-label classification aims at learning classifiers to associate each item with a set of labels. Yet, different from the crowdsourcing setting, it is based on the features of the data itself, such as image pixels or textual indicators [38]. Ordinal classification studies a similar setting, yet assuming a natural ordering among labels. It can be traced back to multi-label classification through membership functions [18]. Our work considers a more generic relation between labels in terms of their co-occurrence for the items in a cluster. Data streams classification aims at multi-label classification in an online setting, processing data in a real-time manner [10].Despite the differences in the underlying classification problem (answer aggregation is not based on features of the items that shall be labelled), this setting is similar to the online setting in crowdsourcing.

Multi-label problems have been studied in the context of crowdsourcing before, yet the focus has been primarily on minimizing cost when posting tasks, see [5]. Another example is work on optimising the cost of hiring workers when generating training data for classifiers [4]. However, these approaches assume labels to be independent and consider all workers equally—adopting some form of majority voting to aggregate answers. Answer aggregation for multi-label crowdsourcing that takes into account the worker communities, partial answer validity, label dependencies, and adaptivity of the aggregation model, in turn, has not been addressed before.

Bayesian Models. Graphical probabilistic models have been successfully applied in various domains, such as image processing, video encoding, and machine learning [21]. Their main benefit is the ability to explicitly capture dependencies between random variables, which often done by means of factor graphs. However, most attempts to use graphical models for multi-label answer aggregation, e.g., [4], show two two major limitations: (1) the models are parametric, which enforces assumptions on the true distribution of crowdsourcing data, even though there is no ground truth available;

(2) they ignore worker communities, even though spammers may have a huge impact on the aggregation result, see [39].

To cope with the first limitation, Bayesian nonparametric models may be used. Their flexibility in terms of a variable number of model parameters makes them well suited to characterise the distributions underlying real-world data. As discussed in §2.3, a first application of Bayesian nonparametric models to answer aggregation for single-label crowdsourcing was recently proposed by Moreno et al. [20]. In this work, we generalised these ideas and presented the cBCMC model for multi-label problems in crowdsourcing. Unlike the existing work, our model supports partial answer validity and exploits co-occurrence dependencies between labels.

7. CONCLUSION AND FUTURE WORK

In this paper, we presented a novel Bayesian nonparametric approach to aggregate multi-label crowdsourcing answers. The key features of the proposed cBCMC model are its ability to capture worker characteristics (by worker communities) and dependencies between the labels assigned to items (by item clusters). The former ultimately improves precision by separating answers of faulty workers from those of reliable workers; the latter ultimately improves recall by exploiting co-occurrence dependencies to complete results. We further presented inference and prediction mechanisms for the cBCMC model for both, static as well as online scenarios. Our experimental results showed that answer aggregation based on the cBCMC model outperforms a baseline method by up to 134% in precision and recall, while being robust against spammers and the answer sparsity as often observed in crowdsourcing. In future work, we intend to lift our model to other types of crowdsourcing tasks, such as ranking or assignment of continuous labels.

References

[1] Q. Abbas et al. "Pattern classification of dermoscopy images: A perceptually uniform model". In: *JPR* (2013), pp. 86–97.

[2] Y. Amsterdamer et al. "Crowd Mining". In: *SIGMOD*. 2013, pp. 241–252.

[3] D. Blei et al. "Variational inference for Dirichlet process mixtures". In: *Bayesian Anal.* (2006), pp. 121–143.

[4] J. Bragg et al. "Crowdsourcing multi-label classification for taxonomy creation". In: *HCOMP*. 2013.

[5] C. C. Cao et al. "Whom to ask?: jury selection for decision making tasks on micro-blog services". In: *VLDB*. 2012, pp. 1495–1506.

[6] T.-S. Chua et al. "NUS-WIDE: a real-world web image database from National University of Singapore". In: *CIVR*. 2009, p. 48.

[7] A. P. Dempster et al. "Maximum likelihood from incomplete data via the EM algorithm". In: *J. R. Stat. Soc* (1977), pp. 1–38.

[8] D. E. Difallah et al. "Mechanical Cheat: Spamming Schemes and Adversarial Techniques on Crowdsourcing Platforms." In: *CrowdSearch*. 2012, pp. 26–30.

[9] T. Ferguson. "A Bayesian analysis of some nonparametric problems". In: *AOS* (1973), pp. 209–230.

[10] M. M. Gaber et al. "A survey of classification methods in data streams". In: *Data Streams*. 2007, pp. 39–59.

[11] G. Ganu et al. "Beyond the Stars: Improving Rating Predictions using Review Text Content." In: *WebDB*. Vol. 9. 2009, pp. 1–6.

[12] C. Gokhale et al. "Corleone: Hands-off Crowdsourcing for Entity Matching". In: *SIGMOD*. 2014, pp. 601–612.

[13] M. D. Hoffman et al. "Stochastic variational inference". In: *JMLR* (2013), pp. 1303–1347.

[14] J. J. Horton et al. "The labor economics of paid crowdsourcing". In: *EC*. 2010, pp. 209–218.

[15] N. Q. V. Hung et al. "An Evaluation of Aggregation Techniques in Crowdsourcing". In: *WISE*. 2013, pp. 1–15.

[16] M. Jordan et al. "An introduction to variational methods for graphical models". In: *ML* (1999), pp. 183–233.

[17] G. Kazai et al. "Worker types and personality traits in crowdsourcing relevance labels". In: *CIKM*. 2011, pp. 1941–1944.

[18] W. Kotlowski et al. "On nonparametric ordinal classification with monotonicity constraints". In: *TKDE* (2013), pp. 2576–2589.

[19] K. Lee et al. "The social honeypot project: protecting online communities from spammers". In: *WWW*. 2010, pp. 1139–1140.

[20] P. G. Moreno et al. "Bayesian Nonparametric Crowdsourcing". In: *JMLR* (2015), pp. 1607–1627.

[21] K. P. Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[22] I. Ounis et al. "Overview of the trec-2011 microblog track". In: *TREC*. 2011.

[23] J. Pavlopoulos et al. "Aspect term extraction for sentiment analysis: New datasets, new evaluation measures and an improved unsupervised method". In: *ACL*. 2014, pp. 44–52.

[24] A. J. Quinn et al. "Human computation: a survey and taxonomy of a growing field". In: *CHI*. 2011, pp. 1403–1412.

[25] V. C. Raykar et al. "Ranking annotators for crowdsourced labeling tasks". In: *NIPS*. 2011, pp. 1809–1817.

[26] A. Ritter et al. "Named entity recognition in tweets: an experimental study". In: *ACL*. 2011, pp. 1524–1534.

[27] J. Ross et al. "Who are the crowdworkers?: shifting demographics in mechanical turk". In: *CHI*. 2010, pp. 2863–2872.

[28] S. Sen et al. "Tagging, communities, vocabulary, evolution". In: *CSCW*. 2006, pp. 181–190.

[29] J. Sethuraman. "A constructive definition of Dirichlet priors". In: *Statistica Sinica* (1994), pp. 639–650.

[30] C. Sun et al. "Chimera: Large-Scale Classification using Machine Learning, Rules, and Crowdsourcing". In: *VLDB*. 2014, pp. 1529–1540.

[31] `http://www.crowdflower.com/`.

[32] J. Vuurens et al. "How much spam can you take? an analysis of crowdsourcing results to increase accuracy". In: *CIR*. 2011, pp. 48–55.

[33] C. Wang et al. "Online variational inference for the hierarchical Dirichlet process". In: *AISTATS*. 2011.

[34] F. L. Wauthier et al. "Bayesian bias mitigation for crowdsourcing". In: *NIPS*. 2011, pp. 1800–1808.

[35] T. Yan et al. "CrowdSearch: Exploiting Crowds for Accurate Real-time Image Search on Mobile Phones". In: *MobiSys*. 2010, pp. 77–90.

[36] J. Yi et al. "Semi-crowdsourced clustering: Generalizing crowd labeling by robust distance metric learning". In: *NIPS*. 2012, pp. 1772–1780.

[37] C. J. Zhang et al. "Reducing Uncertainty of Schema Matching via Crowdsourcing". In: *VLDB*. 2013, pp. 757–768.

[38] M.-L. Zhang et al. "Multi-label learning by exploiting label dependency". In: *KDD*. ACM. 2010, pp. 999–1008.

[39] B. Zhao et al. "A bayesian approach to discovering truth from conflicting sources for data integration". In: *VLDB*. 2012, pp. 550–561.

APPENDIX

A. OFFLINE LEARNING WITH DETERMINISTIC VARIATIONAL INFERENCE

We provide the detailed computation of two equations:

$$\kappa_{ium} \propto \exp\left(\sum_{i=1}^{I}\sum_{t=1}^{T}\varphi_{it}\mathbb{E}\left[\ln p\left(x_{iu} \mid \psi_m^t\right)\right] + \mathbb{E}\left[\ln \pi_m\right]\right)$$

$$\varphi_{it} \propto \exp\left(\mathbb{E}\left[\ln p\left(y_i \mid \phi_t\right)\right] + \mathbb{E}\left[\ln \tau_t\right]\right)$$

by:

$$\begin{aligned}\mathbb{E}\left[\ln p\left(x_{iu} \mid \psi\right)\right] = &\sum_{c=1}^{C} x_{iuc}\left(\psi\left(\lambda_{mc}^t\right) - \psi\left(\sum_c \lambda_{mc}^t\right)\right) \\ &+ \ln\Gamma\left(\sum_c x_{iuc} + 1\right) - \sum_c \ln\Gamma\left(x_{iuc} + 1\right)\end{aligned}$$

$$\begin{aligned}\mathbb{E}\left[\ln p\left(y_i \mid \phi_t\right)\right] = &\sum_{c=1}^{C} x_{iuc}\left(\psi\left(\zeta_{tc}\right) - \psi\left(\sum_c \zeta_{tc}\right)\right) \\ &+ \ln\Gamma\left(\sum_c y_{ic} + 1\right) - \sum_c \ln\Gamma\left(y_{ic} + 1\right)\end{aligned}$$

$$\begin{aligned}
&\mathbb{E}\left[\ln \pi_m\right] = \mathbb{E}\left[\ln \pi'_m\right] + \sum_{k=1}^{m-1}\mathbb{E}\left[\ln\left(1-\pi'_k\right)\right] \\
&\mathbb{E}\left[\ln \pi'_m\right] = \psi\left(\rho_{m1}\right) - \psi\left(\rho_{m1}+\rho_{m2}\right) \\
&\mathbb{E}\left[\ln\left(1-\pi'_m\right)\right] = \psi\left(\rho_{m2}\right) - \psi\left(\rho_{m1}+\rho_{m2}\right) \\
&\mathbb{E}\left[\ln \tau_t\right] = \mathbb{E}\left[\ln \tau'_t\right] + \sum_{k=1}^{t-1}\mathbb{E}\left[\ln\left(1-\tau'_k\right)\right] \\
&\mathbb{E}\left[\ln \tau'_t\right] = \psi\left(\upsilon_{t1}\right) - \psi\left(\upsilon_{t1}+\upsilon_t\right) \\
&\mathbb{E}\left[\ln\left(1-\tau'_t\right)\right] = \psi\left(\upsilon_{m2}\right) - \psi\left(\upsilon_{m1}+\upsilon_{m2}\right)
\end{aligned}$$

Here, $\psi(\cdot)$ is digamma function; $\Gamma(\cdot)$ is gamma function.

B. ONLINE LEARNING WITH STOCHASTIC VARIATIONAL INFERENCE

In this online setting, we suppose that we have the number of workers and the number of answers from workers will be increased by time. The ELBO function that we need to optimize become:

$$\mathcal{L} = \mathbb{E}\left[\ln p\left(\pi', \tau', z, l, \psi, \phi, x, y\right)\right] - \mathbb{E}\left[\ln q\left(\pi', \tau', z, l, \psi, \phi\right)\right]$$

$$
\begin{aligned}
\mathbb{E}\left[\ln p\left(\pi',\tau',z,l,\psi,\phi,x,y\right)\right] &= \mathbb{E}\left[\ln p\left(x \mid \pi',\tau',z,l,\psi,\phi\right)\right] + \mathbb{E}\left[\ln p\left(y \mid \tau',l,\psi,\phi\right)\right] \\
&= \sum_{u,i}\mathbb{E}\left[\ln p\left(x_{ui} \mid \psi_{z_u l_i}\right)\right] + \sum_{u}\mathbb{E}\left[\ln p\left(z_u \mid \pi'\right)\right] \\
&\quad + \sum_{i}\mathbb{E}\left[\ln p\left(y_i \mid \phi_{l_i}\right)\right] + \sum_{i}\mathbb{E}\left[\ln p\left(l_i \mid \tau'\right)\right] \\
&\quad + \sum_{m,t}\mathbb{E}\left[\ln p\left(\psi_m^t\right)\right] + \sum_{t}\mathbb{E}\left[\ln p\left(\phi_t\right)\right] \\
&\quad + \sum_{m=1}^{M-1}\mathbb{E}\left[\ln p\left(\pi_m^{'}\right)\right] + \sum_{t=1}^{T-1}\mathbb{E}\left[\ln p\left(\tau_t^{'}\right)\right] \\
&= \sum_{u}\left(\sum_{i}\mathbb{E}\left[\ln p\left(x_{ui} \mid \psi_{z_u l_i}\right)\right] + \mathbb{E}\left[\ln p\left(z_u \mid \pi'\right)\right]\right) \\
&\quad + \sum_{i}\mathbb{E}\left[\ln p\left(y_i \mid \phi_{l_i}\right)\right] + \sum_{i}\mathbb{E}\left[\ln p\left(l_i \mid \tau'\right)\right] \\
&\quad + \sum_{m,t}\mathbb{E}\left[\ln p\left(\psi_m^t\right)\right] + \sum_{t}\mathbb{E}\left[\ln p\left(\phi_t\right)\right] \\
&\quad + \sum_{m=1}^{M-1}\mathbb{E}\left[\ln p\left(\pi_m^{'}\right)\right] + \sum_{t=1}^{T-1}\mathbb{E}\left[\ln p\left(\tau_t^{'}\right)\right]
\end{aligned}
$$

$$
\begin{aligned}
\mathbb{E}\left[\ln q\left(\pi',\tau',z,l,\psi,\phi\right)\right] &= \sum_{u}\mathbb{E}\left[\ln q\left(z_u\right)\right] + \sum_{i}\mathbb{E}\left[\ln q\left(l_i\right)\right] + \sum_{m,t}\mathbb{E}\left[\ln q\left(\psi_m^t\right)\right] \\
&\quad + \sum_{t}\mathbb{E}\left[\ln q\left(\phi_t\right)\right] + \sum_{m=1}^{M-1}\mathbb{E}\left[\ln q\left(\pi_m^{'}\right)\right] + \sum_{t=1}^{T-1}\mathbb{E}\left[\ln q\left(\tau_t^{'}\right)\right]
\end{aligned}
$$

Therefore, the global parameters now include $q\left(\psi_m^t \mid \lambda_m^t\right)$, $q\left(\phi_t \mid \zeta_t\right)$, $q\left(l_i \mid \varphi_i\right)$, $q\left(\pi' \mid \rho\right)$ and $q\left(\tau' \mid \upsilon\right)$. Their natural gradients are:

$$
\begin{aligned}
\frac{\partial \mathcal{L}_u}{\partial \lambda_m^t} &= \frac{\partial}{\partial \lambda_m^t}\left(\sum_{i}\mathbb{E}\left[\ln p\left(x_{ui} \mid \psi_{z_u l_i}\right)\right] + \frac{1}{U}\mathbb{E}\left[\ln p\left(\psi_m^t\right)\right] - \frac{1}{U}\mathbb{E}\left[\ln q\left(\psi_m^t\right)\right]\right) \\
\frac{\partial \mathcal{L}_u^{ng}}{\partial \lambda_m^t} &= \frac{-\lambda_m^t + \gamma_m^t + U\sum_i \varphi_{it}\kappa_{um}\left[T\left(x_j\right),1\right]}{U}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}_u}{\partial \zeta_t} &= \frac{\partial}{\partial \zeta_t}\left(\frac{1}{U}\sum_{i}\mathbb{E}\left[\ln p\left(y_i \mid \phi_{l_i}\right)\right] + \frac{1}{U}\mathbb{E}\left[\ln p\left(\phi_t\right)\right] - \frac{1}{U}\mathbb{E}\left[\ln q\left(\psi_m^t\right)\right]\right) \\
\frac{\partial \mathcal{L}_u^{ng}}{\partial \zeta_t} &= \frac{-\zeta_t + \eta + \sum_i \varphi_{it}\left[T\left(y_i\right),1\right]}{U}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}_u}{\partial \rho_{m1}} &= \frac{\partial}{\partial \rho_{m1}}\left(\mathbb{E}\left[\ln p\left(z_u \mid \pi'\right)\right] + \frac{1}{U}\mathbb{E}\left[\ln p\left(\pi_m^{'}\right)\right] - \frac{1}{U}\mathbb{E}\left[\ln q\left(\pi_m^{'}\right)\right]\right) \\
\frac{\partial \mathcal{L}_u^{ng}}{\partial \rho_{m1}} &= \frac{-\rho_{m1} + 1 + U\kappa_{um}}{U} \\
\frac{\partial \mathcal{L}_u}{\partial \rho_{m2}} &= \frac{\partial}{\partial \rho_{m2}}\left(\mathbb{E}\left[\ln p\left(z_u \mid \pi'\right)\right] + \frac{1}{U}\mathbb{E}\left[\ln p\left(\pi_m^{'}\right)\right] - \frac{1}{U}\mathbb{E}\left[\ln q\left(\pi_m^{'}\right)\right]\right) \\
\frac{\partial \mathcal{L}_u^{ng}}{\partial \rho_{m2}} &= \frac{-\rho_{m2} + 1 + U\sum_{l=m+1}^{M}\kappa_{um}}{U}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}_u}{\partial \upsilon_{t1}} &= \frac{\partial}{\partial \upsilon_{t1}}\left(\frac{1}{U}\sum_{i}\mathbb{E}\left[\ln p\left(l_i \mid \tau'\right)\right] + \frac{1}{U}\mathbb{E}\left[\ln p\left(\tau_t^{'}\right)\right] - \frac{1}{U}\mathbb{E}\left[\ln q\left(\tau_t^{'}\right)\right]\right) \\
\frac{\partial \mathcal{L}_u^{ng}}{\partial \upsilon_{t1}} &= \frac{-\upsilon_{t1} + 1 + \sum_i \varphi_{it}}{U}
\end{aligned}
$$

$$\frac{\partial \mathcal{L}_u}{\partial \upsilon_{t2}} = \frac{\partial}{\partial \upsilon_{t2}} \left(\frac{1}{U} \sum_i \mathbb{E}\left[\ln p\left(l_i \mid \tau'\right)\right] + \frac{1}{U} \mathbb{E}\left[\ln p\left(\tau_t'\right)\right] - \frac{1}{U} \mathbb{E}\left[\ln q\left(\tau_t'\right)\right] \right)$$

$$\frac{\partial \mathcal{L}_u^{ng}}{\partial \upsilon_{t2}} = \frac{-\upsilon_{t2} + 1 + \sum_{l=t+1}^{T} \sum_i \varphi_{il}}{U}$$

The natural gradient for the canonical parameter μ of mean parameter φ is:

$$\begin{aligned}
\frac{\partial \mathcal{L}_u}{\partial \mu_i} &= \frac{\partial}{\partial \mu_i} \left(\mathbb{E}\left[\ln p\left(x_{ui} \mid \psi_{z_u l_i}\right)\right] + \frac{1}{U} \sum_i \mathbb{E}\left[\ln p\left(y_i \mid \phi_{l_i}\right)\right] - \frac{1}{U} \sum_i \mathbb{E}\left[\ln q\left(l_i\right)\right] \right) \\
&= \frac{\partial}{\partial \mu_i} \left(\mathbb{E}\left[\ln p\left(x_{ui} \mid \psi_{z_u l_i}\right)\right] + \frac{1}{U} \sum_i \mathbb{E}\left[\ln p\left(y_i \mid \phi_{l_i}\right)\right] - \frac{1}{U} \sum_i \mathbb{E}\left[\ln q\left(l_i\right)\right] \right)
\end{aligned}$$

$$\begin{aligned}
\frac{\partial}{\partial \mu_i} \mathbb{E}\left[\ln p\left(x_{iu} \mid \psi_{z_u l_i}\right)\right] &= \frac{\partial}{\partial \mu_i} \sum_t \left(\sum_m \kappa_{um} \mathbb{E}\left[\ln p\left(x_{iu} \mid \psi_m^t\right)\right] \right) \varphi_{it} \\
&= \left[a_{i1} - a_{1T}, \ldots, a_{i(T-1)} - a_{(T-1)T}\right] \frac{\partial^2 B\left(\mu_i\right)}{\partial \mu_i \partial \mu_i^\top}
\end{aligned}$$

Therefore:

$$\frac{\partial \mathcal{L}_u^{ng}}{\partial \mu_{it}} = \frac{-\mu_{it} + \mathbb{E}\left[\varepsilon_t\right] - \mathbb{E}\left[\varepsilon_T\right] + U\left(a_{it} - a_{tT}\right)}{U}$$

where $a_{it} = \sum_m \kappa_{um} \mathbb{E}\left[\ln p\left(x_{iu} \mid \psi_m^t\right)\right] \quad t = 1, \ldots, T-1$.

C. PREDICTION

We have:

$$\begin{aligned}
p\left(y_i, \boldsymbol{x}_{U_i} \mid \mathcal{D}, \mathcal{P}\right) &= \int p\left(y_i, \boldsymbol{x}_{U_i} \mid \Theta\right) p\left(\Theta \mid \mathcal{D}, \mathcal{P}\right) d\Theta \\
&\approx \int p\left(y_i, \boldsymbol{x}_{U_i} \mid \Theta\right) q\left(\Theta\right) d\Theta \quad \text{(from Variational Bayesian in the inference step)}
\end{aligned}$$

Therefore, the true input of our prediction step is the approximated posterior variational distributions, $q\left(\psi_m^t \mid \lambda_m^t\right)$, $q\left(\phi_t \mid \zeta_t\right)$, $q\left(\pi' \mid \rho\right)$ and $q\left(\tau' \mid \upsilon\right)$. We can use MAP estimation to obtain values for ψ_m^t and ϕ_t which are denoted $\psi_m^{(t)MAP}$ and ϕ_t^{MAP} from this (approximated) posterior. The cluster proportion distributions, , $q\left(\pi' \mid \rho\right)$ and $q\left(\tau' \mid \upsilon\right)$, are used as the full distributions.

$$\begin{aligned}
\int p\left(y_i, \boldsymbol{x}_{U_i} \mid \Theta\right) q\left(\Theta\right) d\Theta &= \sum_{l_i} \int \prod_{u \in U_i} \sum_{z_u} p\left(x_{ui} \mid z_u, l_i, \Theta\right) p\left(z_u \mid \Theta\right) \\
&\qquad p\left(y_i \mid l_i, \Theta\right) p\left(l_i \mid \Theta\right) q\left(\Theta\right) d\Theta \\
&= \sum_{l_i} p\left(l_i \mid \upsilon\right) \int p\left(y_i \mid l_i, \phi\right) q\left(\phi\right) d\phi \\
&\qquad \prod_{u \in U_i} \sum_{z_u} p\left(z_u \mid \kappa_u\right) \int p\left(x_{ui} \mid z_u, l_i, \psi\right) q\left(\psi\right) d\psi \\
&= \mathbb{E}\left[\mathbb{E}\left[p\left(y_i \mid l_i, \phi\right)\right]_{q(\phi)}\right]_{q(l_i)} \\
&\qquad \prod_{u \in U_i} \mathbb{E}\left[\mathbb{E}\left[p\left(x_{ui} \mid z_u, l_i, \psi\right)\right]_{q(\psi)}\right]_{q(z_u)}
\end{aligned}$$

$$\mathbb{E}\left[\mathbb{E}\left[p\left(y_i \mid l_i, \phi\right)\right]_{q(\phi)}\right]_{q(l_i)} = \sum_{t=1}^{T} \varphi_{it} \mathbb{E}\left[p\left(y_i \mid \phi_t\right)\right] = \sum_{t=1}^{T} \varphi_{it} p\left(y_i \mid \phi_t^{MAP}\right)$$

$$\mathbb{E}\left[\mathbb{E}\left[p\left(x_{ui} \mid z_u, l_i, \psi\right)\right]_{q(\psi)}\right]_{q(z_u)q(l_i)} = \sum_{t=1}^{T} \varphi_{it} \sum_{m=1}^{M} \kappa_{um} \mathbb{E}\left[p\left(x_{ui} \mid \psi_m^t\right)\right] = \sum_{t=1}^{T} \varphi_{it} \sum_{m=1}^{M} \kappa_{um} p\left(x_{ui} \mid \psi_m^{(t)MAP}\right)$$

where ϕ_t^{MAP} and $\psi_m^{(t)MAP}$ are MAP estimates (a.k.a. mode) of their variational distributions $q\left(\phi_t \mid \zeta_t\right) = \mathrm{Dir}(\zeta_t)$, $q\left(\psi_m^t \mid \lambda_m^t\right) = \mathrm{Dir}(\lambda_m^t)$, which are well-known for Dirichlet distributions.